(http://csmatters.org) 6 - 1a

0b110 - 0b1

Data Visualization with EarSketch 2020-2021

Unit 6. Data Visualization

Revision Date: Jan 05, 2020
Duration: 6 50-minute sessions

Lesson Summary

Use EarSketch to review the programming topics learned this year and as a final preparation for the Create Performance Task.

Lesson Outcomes

- Use EarSketch to write programs for personal expression and to facilitate the expressions of others.
- Explain legal and ethical issues related to the use of EarSketch or other computing platforms.
- Use EarSketch to visualize data in the form of music.
- Use data abstractions with EarSketch and explain their benefit to program development.
- Use sequence, selection, and iteration both in pseudocode and in program code.
- Develop procedures that leverage parameters for code reuse and explain their benefit to program development.
- Use program code in existing libraries and explain its benefit to program development.
- Demonstrate how the capabilities of users and their tools impact their ability to process data and the results.

Sessions 1 - 6 are required.

Sessions 7 - 10 can be used as a final practice for the Create Performance Task.

Learning Objectives

CSP Objectives

- *EU CRD-1 - Incorporating multiple perspectives through collaboration improves computing innovations as they are developed.*
    - LO CRD-1.A - Explain how computing innovations are improved through collaboration.
- *EU CRD-2 - Developers create and innovate using an iterative design process that is user-focused, that incorporates implementation/feedback cycles, and that leaves ample room for experimentation and risk-taking.*
    - LO CRD-2.A - Describe the purpose of a computing innovation.
    - LO CRD-2.B - Explain how a program or code segment functions.
    - LO CRD-2.C - Identify input(s) to a program.
    - LO CRD-2.E - Develop a program using a development process.
    - LO CRD-2.F - Design a program and its user interface.
    - LO CRD-2.J - Identify inputs and corresponding expected outputs or behaviors that can be used to check the correctness of an algorithm or program.
- *EU DAT-1 - The way a computer represents data internally is different from the way the data is interpreted and displayed for the user. Programs are used to translate data into a representation more easily understood by people.*
    - LO DAT-1.A - Explain how data can be represented using bits.
- *EU AAP-1 - To find specific solutions to generalizable problems, programmers represent and organize data in multiple ways.*
    - LO AAP-1.A - Represent a value with a variable.
    - LO AAP-1.B - Determine the value of a variable as a result of an assignment.
    - LO AAP-1.C - Represent a list or string using a variable.
    - LO AAP-1.D - For data abstraction: a. Develop data abstraction using lists to store multiple elements. b. Explain how the use of data abstraction manages complexity in program code.
- *EU AAP-2 - The way statements are sequenced and combined in a program determines the computed result. Programs incorporate iteration and selection constructs to represent repetition and make decisions to handle varied input values.*
    - LO AAP-2.A - Express an algorithm that uses sequencing without using a programming language.
    - LO AAP-2.B - Represent a step-by-step algorithmic process using sequential code statements.
    - LO AAP-2.C - Evaluate expressions that use arithmetic operators.
    - LO AAP-2.E - For relationships between two variables, expressions, or values: a. Write expressions using relational operators. b. Evaluate expressions that use relational operators.
    - LO AAP-2.G - Express an algorithm that uses selection without using a programming language.
    - LO AAP-2.H - For selection: a. Write conditional statements. b. Determine the result of conditional statements.
    - LO AAP-2.J - Express an algorithm that uses iteration without using a programming language.
    - LO AAP-2.K - For iteration: a. Write iteration statements. b. Determine the result or side-effect of iteration statements.
    - LO AAP-2.O - For algorithms involving elements of a list: a. Write iteration statements to traverse a list. b. Determine the result of an algorithm that includes list traversals.

- *EU AAP-3 - Programmers break down problems into smaller and more manageable pieces. By creating procedures and leveraging parameters, programmers generalize processes that can be reused. Procedures allow programmers to draw upon existing code that has already been tested, allowing them to write programs more quickly and with more confidence.*
    - LO AAP-3.A - For procedure calls: a. Write statements to call procedures. b. Determine the result or effect of a procedure call.
    - LO AAP-3.B - Explain how the use of procedural abstraction manages complexity in a program.
    - LO AAP-3.D - Select appropriate libraries or existing code segments to use in creating new programs.
    - LO AAP-3.E - For generating random values: a. Write expressions to generate possible values. b. Evaluate expressions to determine the possible results.
    - LO AAP-3.F - For simulations: a. Explain how computers can be used to represent real-world phenomena or outcomes. b. Compare simulations with real-world contexts.

Math Common Core Practice:

- MP2: Reason abstractly and quantitatively.
- MP5: Use appropriate tools strategically.
- MP7: Look for and make use of structure.

Common Core Math:

- A-SSE.1-2: Interpret the structure of expressions
- F-IF.1-3: Understand the concept of a function and use function notation
- F-IF.4-6: Interpret functions that arise in applications in terms of the context
- F-BF.1-2: Build a function that models a relationship between two quantities

Common Core ELA:

- RST 12.3 - Precisely follow a complex multistep procedure
- RST 12.4 - Determine the meaning of symbols, key terms, and other domain-specific words and phrases
- WHST 12.2 - Write informative/explanatory texts, including the narration of historical events, scientific procedures/experiments, or technical processes

NGSS Practices:

- 5. Using mathematics and computational thinking
- 6. Constructing explanations (for science) and designing solutions (engineering)

Teacher Resources

Lesson Plan

Session 1 - Introduction to EarSketch

Lesson Overview

Say: We will be using EarSketch to review the programming topics we learned this year and as a final preparation for before the Create Performance Task. EarSketch is intended as a tool to teach novice programmers and we are no longer novices.

Activity 1

Explore the EarSketch DAW, IDE, Curriculum, API and Sound Library

- The EarSketch Library is an example of collections of code that can be used once imported into the program.
- The EarSketch API is an example of a programmer interface which enables programmers to use functions defined in the library.
- Libraries make constructing complex programs more managable.

EarSketch Teacher Presentation - Lesson 1

Legal and Ethical Issues

Read Unit 1 Lesson 7.

Is it right for people to own the products of their own labor?

Who owns the music you make in EarSketch?

Why would someone give away their rights as an owner?

What provision of law protects the ownership rights of music and software creators?

Activity 2

First MiniTask Assignment - Same Folder

Make a 4 measure piece of music that has 3 sounds. All three sounds should come from the same artist and genre in the sound browser. Use the constants from the sound browser as parameters for your fitMedia() function calls.

Wrap-up: (4 min)

Use the EarSketch API to find the description of these two functions:

- fitMedia
- setEffect

Session 2 Making Music with EarSketch using sequence, procedural abstraction and iteration

Getting Started

Unit 1 Summary (5 min)

Jigsaw the 8 sections of the Unit 1 Summary in EarSketch. Have each group discuss - 1 min- and then share their section in their own words. Each presentation has a 30 second time limit.

Activity 1 (20 min)

Introduction

Say: you have used plenty of functions. However, in EarSketch you aren't limited to using the functions within the EarSketch API, like fitMedia() and setEffect(). You can make new functions to do anything you want. These custom functions allow you to group together lines of code to form a single instruction for the computer. This process is known as abstraction.

This session focuses on making music using sequence and iteration, and coding large-scale changes in music efficiently. Several measures that express an idea or feeling make up a song section. Songs that contain multiple sections allow for variety and structure, or form. Intros, Verses, Choruses, and Outros are examples of sections that contribute to form.

Edit the script in section 9.2: A-B-A Form.

Say: The code is somewhat messy, and a little confusing. Code is also repeated for the second sectionA, an indication that the program could be written more efficiently. We will improve this code using a custom function.

Consider the following code. Note the definition of the function myFunction and the call of myFunction. We will define and call custom functions to improve the A-B-A form example.

```
# python code
#
# script_name: Custom Functions
#
# author: The EarSketch Team
#
# description: Defining our own function that makes a section of music
#
#
```

```
#
#Setup
from earsketch import *
init()
setTempo(100)

#Music

# Defining our new function with two parameters
def myFunction(startMeasure, endMeasure):
  fitMedia(ELECTRO_DRUM_MAIN_BEAT_003, 1, startMeasure, endMeasure)
  fitMedia(ELECTRO_ANALOGUE_PHASERBASS_003, 2, startMeasure, endMeasure)

# Calling our function, passing it two arguments: 1 and 17.
myFunction(1, 17)

#Finish
finish()
```

Say: Move Section A code into two functions named sectionA and SectionB. Use parameter startMeasure and endMeasure for each function. Delete the second sectionA code. Call the functions using the arguments in the code snippet below. Note that sectionA is used twice but only written once. Reuse of code is one of the advantages of procedural abstraction.

```
# Setting up an ABA musical form through function calls

sectionA(1, 5)

sectionB(5, 9)

sectionA(9, 13)
```

Examine the A-B-A-B form WITHOUT functions and compare it to the A-B-A-B Form WITH functions.

Think pair share: What benefit in addition to code reuse are functions to program development?

Activity 2 (20 Min)

These examples have used sequential statements and functions. Next we will use looping but first we need to examine the EarSketch function makeBeat. Watch the first three minutes of this video about makeBea (https://earsketch.gatech.edu/earsketch2/videoMedia/011-03-makeBeat()-

PY.mp4)t.

Edit the Multi Beat script in EarSketch lesson 11.3. Run and play the music. Modify the beat strings beat 1 and beat2. Run and play the revised beats.

Remind students about use of the makebeat() function.

Say:

- Strings are used with the makeBeat() function to create rhythmic patterns in EarSketch. Function makeBeat() uses a beat string parameter to define each sixteenth note sub-beat of its pattern. A 0 starts playing a clip, a + extends the note for the next sub-beat, and - creates a rest.
- makeBeat() takes four arguments:
    - **clipName:** The clip a beat is constructed from.
    - **trackNumber:** The track on which music is placed.
    - **measureNumber:** The starting measure of the beat. The beat string determines the total length.
    - **beatString:** A string that specifies the rhythm created.

Edit the MultBeat script in Lesson 12.2. Create two additional beat string variables and a second for loop. The second for loop is to use the two new beat strings for tracks 1 and 2 but measures 5 - 8.

Compare the Drum beat (no loops) code in Lesson 12.3 to the Drum beat (with loops) code.

Think-pair-share: Discuss with your elbow partner at least two benefits of using looping to program development.

Session 3 Visualizing Music with EarSketch 1

Warm Up (3 min)

Say: In this session, we're going to first review how colors are represented by the computer, then show ways you can create visuals that react to the music you have been composing.

Journal: How would you explain to an elementary school student how pictures are stored on a cell phone?

Activity 1 (15 min)

Introduction

Say: Visualizations can be created from any data such as music. After these tutorials you will be able to make visuals like the one shown below:

https://earsketch.gatech.edu/earsketch2/videoMedia/028-01-InitialSteps-PY-JS.mp4 (https://earsketch.gatech.edu/earsketch2/videoMedia/028-01-InitialSteps-PY-JS.mp4)

Discussion: Which do you think has more impact on the ability to create something of value using data: the capabilities of the user, or the capabilities of the tool they use?

Say: We'll start by making a basic animation (shown below), that will help us learn the skills to make more complex visuals later on.

https://earsketch.gatech.edu/earsketch2/videoMedia/026-01-RGBColorandHexNotation-PY-JS.mp4 (https://earsketch.gatech.edu/earsketch2/videoMedia/026-01-RGBColorandHexNotation-PY-JS.mp4)

Say: To make computer visuals we need to first learn three concepts. Firstly, how computers understand colors. Then we'll at how computers understand a position in a visual. Finally we'll look at how we draw into these colors and positions.

Present the following:

RGB Color and Hex Notation

You may already know colors can be created by putting different colors together. This can happen with paint when you combine two colors, such as red and blue to make purple. Computers think of all colors by how much of red, green and blue makes up the color. This is then stored as an RGB (red, green, blue) value. RGB values are between 0 (not used at all) to 255 (full intensity). RGB values do not always have the same results as you might think from using paint, so you may want to check the table below.

RGB Combination	Name of Color	Color
255, 255, 255	White	
0, 0, 0	Black	
127, 127, 127	Grey	
255, 0, 0	Red	
0, 255, 0	Green	
0, 0, 255	Blue	
255, 0, 255	Magenta	
255, 255, 0	Yellow	
0, 255, 255	Cyan	

The values for yellow can be the hardest to guess. To help choose colors EarSketch has a color picker tool. This lets you choose a color and then gives you the hex value underneath the color. Try moving the slider to see the values change.

Find the hex number for your favorite color.

A hex value is the way the computer represents the numbers, in this case the three RGB values. Hex notation is created by counting differently. Instead of going 0-9, we instead count 0-F, meaning in each space we can represent more numbers. The table below shows some examples.

Hex	Decimal
7	7

A 10 (A is one greater than 9)

C 12 (Counting in hex: 9, A, B, C)

2D 45 (2D is equal to 2*16, plus 13)

F1 241 (F1 is equal to 15*16, plus 1)

The last shows why hex notation is so useful for colors. We can use two digits to show a value that normally takes three digits (in decimal). Below are some more color examples.

Color	Hex	Decimal (x3)
White	"#FFFFFF"	255, 255, 255
Red	"#FF0000"	255, 0, 0
Grey	"#A0A0A0"	160, 160, 160
Yellow	"#FFFF00"	255, 255, 0

Say: In EarSketch, you can find color number values with the color picker. EarSketch also includes rgbToHex() which changes three numbers to its hex value so your program can define colors.

Activity 2 (15 min)

Basic Drawing with EarSketchdrawRectangle()

For the next section we will need some music in our project to generate visuals. If you have a song prepared already, load it up. If not, use this one or quickly make your own:

```
from earsketch import *

init()


setTempo(120)


fitMedia(EIGHT_BIT_ANALOG_DRUM_LOOP_001,1,1,3)

fitMedia(EIGHT_BIT_ATARI_LEAD_005,2,1,3)

fitMedia(EIGHT_BIT_ATARI_LEAD_008,3,1,3)

fitMedia(EIGHT_BIT_ATARI_LEAD_005,2,3,5)

fitMedia(EIGHT_BIT_ATARI_LEAD_008,3,3,5)


finish()
```

Near the end of this script (just before finish()), we're going to add a call to a new function: drawRectangle(). It takes five arguments: x location, y location, width, height, and color.

As you may remember, a function allows us to abstract a more complex process into a single line of code. This is what drawRectangle() does: it takes the information for where and how to draw the shape we want and--"under the hood"--it does the more complex work of actually changing the canvas.

In order to use drawRectangle() to draw, we also need to call it from inside a special function for visualization in EarSketch called onLoop(). onLoop() is a special function that must be included in all EarSketch scripts that use visualization. We'll learn more about how it works in the next unit.

For now, add the following above finish():

```
from earsketch import *

init()


setTempo(120)


fitMedia(EIGHT_BIT_ANALOG_DRUM_LOOP_001,1,1,3)

fitMedia(EIGHT_BIT_ATARI_LEAD_005,2,1,3)

fitMedia(EIGHT_BIT_ATARI_LEAD_008,3,1,3)

fitMedia(EIGHT_BIT_ATARI_LEAD_005,2,3,5)

fitMedia(EIGHT_BIT_ATARI_LEAD_008,3,3,5)


def onLoop():

  drawRectangle(10,15,50,30,"#FF0000")


finish()
```

Run the program.

Say: When you hit play, you should see a red rectangle appear near the top-left corner of your canvas. It's near (but not quite touching) that corner because the x- and y-coordinates we gave it (as the first two arguments) are 10 and 15, respectively.

The x-coordinate represents the shape's horizontal (left-right) position (in pixels), with greater values corresponding to further locations to the right. The y-coordinate represents the shape's vertical (up-down) position (in pixels), with greater values corresponding to further locations down.

In other words, x- and y-coordinates increase as you go down and to the right, and 0,0 represents the top-left corner.

Assign the following modifications and predict how the visualization will appear then run your program..

- Change the arguments to the draw rectangle function so the rectangle is taller than it is wide and so that its color is yellow.

- Replace the drawRectangle() function argument list with: (0,0,getCanvasWidth(), getCanvasHeight(), "#FF0000")

Say:

Let's now make a major change to our code. Copy the following code sample from EarSketch lesson 26.3. Your code should now look like the code below.

```
from earsketch import *

init()


setTempo(120)


fitMedia(EIGHT_BIT_ANALOG_DRUM_LOOP_001,1,1,3)

fitMedia(EIGHT_BIT_ATARI_LEAD_005,2,1,3)

fitMedia(EIGHT_BIT_ATARI_LEAD_008,3,1,3)

fitMedia(EIGHT_BIT_ATARI_LEAD_005,2,3,5)

fitMedia(EIGHT_BIT_ATARI_LEAD_008,3,3,5)


def onLoop():

 pass #empty function for now


def onMeasure():

 drawRectangle(0,0,getCanvasWidth(),getCanvasHeight(),rgbToHex(255,0,0))


finish()
```

Students run the program.

Ask:

1. Does the rectangle appear?
2. What do you have to do to see the rectangle?

Say: The drawRectangle() call is now inside the onMeasure() function, which is called by EarSketch when your song is playing. It is called automatically at the very beginning of each measure. Press play to see the rectangle appear. Note that this is somewhat different from the behavior of

EarSketch that you've experienced before using visualization: earlier, all of your code ran when you clicked Run. Now, with functions like onMeasure(), you can write code that executes repeatedly as your song is playing.

While your song is playing with this particular definition of onMeasure(), a new rectangle is actually being re-drawn every measure. However, because it's being drawn in the same position with the same size and color, there is no change for us to see.

Activity 3 (15 min)

Say: We'll now adjust our definition of onMeasure() to have a visible change on every measure, in rhythm with the music. In other words, we're going to create our first animation. and we're going to do this in EarSketch by using a variable as an argument for our drawRectangle() call. We'll then change that variable's value over time so that the information used by drawRectangle() changes and gives us different results.

Assign the following modifications. The revised code should look like the following script. Changes are in bold.

- declare a global variable called r outside of our onMeasure() function.
- instead of having the number 255 as the red component of our rgbToHex() call, use the variable just declared.

```
def onLoop():
  pass #empty function for now

r = 255
def onMeasure():
  global r
  drawRectangle(0,0,getCanvasWidth(),getCanvasHeight(),rgbToHex(r,0,0))

finish()
```

Say:If you ran and played this code, you would see the same result as before because we didn't change the value of r yet. Let's make that happen by subtracting 50 from r every measure.

Assign the following modifications to the onMeasure function. Changes are in bold.

- Add the decrement of variable r
- add an if statement to rest r to 2555 if r becomes negative

```
def onMeasure():
  global r
  drawRectangle(0,0,getCanvasWidth(),getCanvasHeight(),rgbToHex(r,0,0))
  r = r-50
 if (r < 0):
```

r = 255

Run and play your program.

Say: As your song plays, you should see the red rectangle getting darker on every new measure, until it resets to fully bright red (due to the if-statement in the code above that resets r to 255 if it drops below 0) and begins dimming again.

Add global variables g and b in start them at zero. Increase them each time the onMeasure function is called by 60 and have them reset to 0 whenever either are greater than 255.

Compare your finished code to the last script from EarSketch lesson 26.3.

Wrap up (2 min)

Explain the purpose of the drawRectangle() and the onMeasure() functions.

Session 4 Visualizing Music with EarSketch 2 Using Data Abstractions

Warm up: (3 min)

Review this list of topics from the previous session. What would you append to the list that we also learned?

- Computers store colors as amounts of red, green and blue (RGB).
- Hex values are a good way to store RGB values as they allow you to store colors using less digits.
- onLoop() must be used for all scripts in EarSketch that create visualizations.
- Canvas height and canvas width are the size of the frame the visualization is created in.

Activity 1 (15 min)

View this video showing an animation that changes as the amplitude of the sound changes.

https://earsketch.gatech.edu/earsketch2/videoMedia/027-01-DefiningonLoop()andGettingAudioData-PY-JS.mp4 (https://earsketch.gatech.edu/earsketch2/videoMedia/027-01-DefiningonLoop()andGettingAudioData-PY-JS.mp4)

Say: In the previous session, you learned about using onMeasure(), a function that you define which is then called regularly by EarSketch. There are other functions that you can define in a similar way. These are also called regularly, but even more often. Two examples are onHalf() and onQuarter(). onHalf() is called on the 1st and 3rd beat of every measure (at a rate of half notes) and onQuarter() is called on every beat (or at a rate of quarter notes).

Another example is **onLoop(), which is called about 60 times per second**. Because this is such a rapid rate, it was important for us to begin with the slower onMeasure() so that we could see how things change easily. We can now move on to using the faster onLoop() to create animations appear to move naturally.

Examine the first EarSketch script in lesson 27.1.

Say: We've defined onLoop(), which is called nearly 60 times per second. The first three lines of code store the value returned by getAmplitude(). As a song is playing, any getAmplitude() call will retrieve the amplitude (loudness between 0 and 1, 0 being silent) at the current time for the track number given as an argument. Calling getAmplitude() often will allow us to use data that is up-to-date to the fraction of a second.

Load and run the first script in EarSketch lesson 27.2

Say:

First, to give ourselves a fresh canvas each time onLoop() runs we draw a black rectangle that is the size of the entire canvas. By drawing a new, slightly different image on every call of onLoop(), we can create the illusion of fluid motion for whatever else we're drawing. In this case, we're drawing bars whose size increases along with the amplitude of their respective track.

We give each rectangle a different y-position, keeping this value relative to the height of the canvas; this allows us to resize the canvas and have our image scale with it correctly. The third argument for each of these drawRectangle() calls is where our data from getAmplitude() (amp values are between 0 and 1) from their corresponding tracks by the width of the canvas. This means that a louder moment from the track will result in a longer bar.

Finally, we give each bar a height relative to the height of the canvas and give each bar a distinct color. Below is a screenshot that also shows the arguments used for each shape (after the math has been done on values like amp1). Discuss with your elbow partner how you would change the code to accomblish each of the following.

- add another audio track and fit another bar on the canvas.
- make bars grow from the bottom of the canvas towards the top (rather than from the left towards the right).

Activity 2 (15 min) - Randomness and transparency

Watch this video to see how we will update our animation. https://earsketch.gatech.edu/earsketch2/videoMedia/028-01-InitialSteps-PY-JS.mp4 (https://earsketch.gatech.edu/earsketch2/videoMedia/028-01-InitialSteps-PY-JS.mp4)

Say: In order to draw the various circles to the canvas, we're going to need to keep track of where they should be. To do this we're going to create lists of x-coordinates and y-coordinates for each color of circle and define a function that will use these lists to set the locations of all of these circles. Examine the first EarSketch script in lesson 28.2.

After students load and preview the EarSketch script in lesson 28.2.

Say:

Lines 13 - 18 create empty lists.

LIne 20 initializes the global variable circlesPerColor as 5.

Lines 21 - 30 define the setCirclePositions() function. The function sets random values for the X and Y values of each circle, so that we get nicely unpredictable results. This means as we don't have to manually enter all of these values. We do this by looping through the lists we set up earlier, assigning a random value to each element of each list.

To ensure that each circle is not drawn off-screen, we multiply each random value (which is between 0 and 1) by the width or height of the canvas (for the x and y values respectively).

Line 33 calls the setCirclePositions() function at the start of every measure.

Load and run the script in EarSketch lesson 28.3.

Assign the following modifications to the script in EarSketch lesson 28.3.

- Modify the assignment statements in the setCirclePositions function to use the list append function.
- Add a few more measures of music by adding additional fitmedia() function calls.

Student code should look like the following.

```
from math import *


from earsketch import *

from random import *

init()


setTempo(120)


fitMedia(EIGHT_BIT_ANALOG_DRUM_LOOP_001,1,1,3)

fitMedia(EIGHT_BIT_ATARI_LEAD_005,2,1,3)

fitMedia(EIGHT_BIT_ATARI_LEAD_008,3,1,3)

fitMedia(EIGHT_BIT_ATARI_LEAD_005,2,3,5)

fitMedia(EIGHT_BIT_ATARI_LEAD_008,3,3,5)

fitMedia(EIGHT_BIT_ANALOG_DRUM_LOOP_001,1,4,7)

fitMedia(EIGHT_BIT_ATARI_LEAD_005,2,5,7)

fitMedia(EIGHT_BIT_ATARI_LEAD_008,3,5,7)

fitMedia(EIGHT_BIT_ATARI_LEAD_005,2,7,9)

fitMedia(EIGHT_BIT_ATARI_LEAD_008,3,7,9)
```

```
rCircleXs = []
gCircleXs = []
bCircleXs = []
rCircleYs = []
gCircleYs = []
bCircleYs = []

circlesPerColor = 5
def setCirclePositions():
 for i in range(0, circlesPerColor):
    #set random x positions
rCircleXs.append(random()*getCanvasWidth())
gCircleXs.append(random()*getCanvasWidth())
bCircleXs.append(random()*getCanvasWidth())
#set random y positions
rCircleYs.append(random()*getCanvasWidth())
gCircleYs.append(random()*getCanvasWidth())
bCircleYs.append(random()*getCanvasWidth())



def onMeasure():
  setCirclePositions()


def onLoop():
global rCircleXs, rCircleYs, gCircleXs, gCircleYs, bCircleXs, bCircleYs
amp1 = getSmoothAmplitude(1)
amp2 = getSmoothAmplitude(2)
amp3 = getSmoothAmplitude(3)
MAX_CIRCLE_SIZE = getCanvasHeight()

drawRectangle(0,0,getCanvasWidth(), getCanvasHeight(), "#000000")
```

```
for i in range(0, len(rCircleXs)):
  drawCircle(rCircleXs[i], rCircleYs[i], MAX_CIRCLE_SIZE*amp1,
rgbaToHex(255,0,0,28+floor(amp1*227)))
for i in range(0, len(gCircleXs)):
  drawCircle(gCircleXs[i], gCircleYs[i], MAX_CIRCLE_SIZE*amp2,
rgbaToHex(0,255,0,28+floor(amp2*227)))


for i in range(0, len(bCircleXs)):
  drawCircle(bCircleXs[i], bCircleYs[i], MAX_CIRCLE_SIZE*amp3,
rgbaToHex(0,0,255,28+floor(amp3*227)))


finish()
```

Activity 3 (10 min)

Run and play the code that's been discussed so far, and take a look at the results. Because we used randomness, your canvas probably won't look exactly like the one in the video, but it will be similar.

Say: Work with your elbow partner to choose and implement at least one of the following changes.

- Using getAmplitude() rather than getSmoothAmplitude()
- Using RGB color rather than RGBA color
- Using a different color for the background (perhaps even one that changes over time)
- Changing the number of circles per track/color
- Changing the size of the circles
- Using different shapes

Wrap up: (5 min)

Think-pair-share: Consider doing this project without using the lists.

1. How would the code be changed?
2. How does the use of lists make the programming process easier?

Session 5 - Using selection with user input in EarSketch

Warm up

Introduction

Say: Soon we will begin the Create Performance Task (Create Task). You do not have to use EarSketch for the Create Task - you may use any tools we have studied or others that you learned elsewhere. You do have to create a project that:

- Is designed for a purpose.
- Uses a data abstraction and a functional abstraction that you implemented.
- Implements an algorithm that uses sequence, selection and iteration.

This session will focus on the piece of the Create Task that we have not reviewed so far - using selection and expressing algorithms using sequence, selection and iteration.

Activity 1 User input and selection

Say: Printing to the console allows information to be displayed to the user. Console Input is text-based data taken from the keyboard, giving a program access to information from the user. Together, printing and console input can be used to allow the user to interact with a program.

In EarSketch, the readInput() function is used for console input. This opens a dialog box asking for input and returns the string typed into the dialog box.

Examine the Simple Console Input script in EarSketch lesson 17.1.

Ask: How does the EarSketch readInput() function differ from the Python input statement?

Load and edit the Musical Console Input script at the end of EarSketch lesson 17.1.

Ask: What is the purpose of the int() function used on line 17?

To make decisions - selections - the computer needs to be able to make comparisons and draw logical conclusions. As a reminder these are the comparison and boolean operators used in Python.

- comparison operators (>, >=, <, <=, ==, !=)
- boolean operators (and, or, not).

Load the Conditionals script from EarSketch lesson 17.3.

Ask:

- What does the program do when the user responds to the prompt with the word “yes”?
- What does the program do when the user responds to the prompt with the words “big bird”?

Examine the Which Comes First script for the end of EarSketch lesson 17.3.

Ask:

- How do the statement blocks differ in the if, elif and else clauses?
- Why might users prefer this program format?

Activity 2 Developing Algorithms

To review building algorithms and expressing them in pseudocode read The building blocks of algorithms. (https://www.khanacademy.org/computing/ap-computer-science-principles/algorithms-101/building-algorithms/a/the-building-blocks-of-algorithms)

Take the four question quiz comparing your results with your elbow partner’s.

Begin planning an earsketch program that:

- Is designed for a purpose.
- Uses a data abstraction and a functional abstraction that you implemented.
- Implements an algorithm that uses sequence, selection and iteration.

This program will be for your own interest. You may do this individually or in collaboration with your elbow partner. You will have the rest of this session to plan the program and all of the next session to implement your plan.

Your plan should include the purpose of the project, identify the procedural and data abstractions you plan to use and include a pseudocode version of at least one algorithm that uses sequence selection and iteration.

Wrap up: (2 min)

Briefly describe your project and submit a copy of your plan either digitally or on paper.

Session 6 Creating your own EarSketch script using input and visualization

Introduction

Say: This session you will build the project you described in the previous sessions. You will have most of this session with the last few minutes dedicated to analyzing the progress and decisions you made.will focus on the piece of the Create Task that we have not reviewed so far - using selection and expressing algorithms using sequence, selection and iteration.

Activity 1: (40 min)

Implement the plan you developed in the previous session. After you get a copy of your plans, begin by making a list of your objectives.

Wrap up: (5 min)

Answer these questions.

Journal:

1. How successful were you in meeting your objectives for today?
2. How did the procedural abstractions (functions you developed) help manage the complexity of the project?
3. How did the data abstractions (functions you developed) help manage the complexity of the project?
4. How was the pseudocode helpful in the development of the project?

Discuss any answers or questions with your elbow partner.

Remaining Sessions are Optional

The remaining sessions can be used as a final practice for the Create Performance Task.

Session 7 Developing algorithms when designing programs for others.

Introduction

Say: This session we will begin a project we will design and build for others. The project must meet the same requirements we built for ourselves. It is to be collaborative - with an elbow partner or partners. - The project may or may not use EarSketch.

Activity 1

Watch this video about Human Centered Design (http://www.designkit.org/human-centered-design).

Complete the project development plan.

Project Development Plan

Identify a project.

Discuss with your group:

People who might benefit from a project you develop.

Brainstorm possible projects.

Select a user or users and a project they would benefit from.

Getto know your users.

Describe who the users are(is) and what the user(s) like or need.

Describe how the user(s) behavior would change if they had your completed project.

Describe the features of your project that would make your project most useful.

Design your project.

Develop user experience goals that align with the desired features and behaviors.

Identify inputs or information your project will require.

Identify outputs the project will produce.

Define a prototype.

Watch this video about prototyping. (https://youtu.be/w7Q6bHVESsE)

Describe a prototype version of your project that seems both something you could do and something you could learn form.

Develop you plan for your prototype.

Identify data structure(s) and function(s) you will use.

For each data structure list:

- Name
- Type
- Description of the values it will hold

For each function list:

- Name
- Parameters
- Return values
- Plan for testing

Identify key algorithm(s) you will implement.

For each algorithm:

1. Describe it using pseudocode or flowcharts.
2. Test the algorithm

Activity 2 - Begin prototype development

Plan prototype development.

Schedule the work to be done.

What order will it be done?

Who will do what parts?

How will the parts be assembled?

When will the prototype testing begin?

Wrap up

Identify any help needed from the teacher.

Submit a copy of the Project Development Plan.

Sessions 8 & 9 Develop the prototype of a program for others.

Introduction

Say: This session is dedicated to the development of your project for others.

The project will be assessed using the latest rubric provided by the College Board. After you complete your project you will also complete a short written report - Project Summary - that includes the following:

Project Summary

Partner Names

Purpose

Data Abstraction

Function in the project

Code showing creation and use.

Explanation of how it helped manage complexity

Functional Abstraction

Function in the project

Code showing definition and calls

Parameter it uses and alternative paths through the procedure.

Explanation of how it helped manage complexity.

Key Algorithm

Purpose of the algorithm

Code and how the algorithm works (pseudocode can be used)

Use of selection and iteration

The latest rubric (updated as of November 2018) is in the lesson folder. The practice project does not ask students to create a video so row 1 should net be used. The assessment will be the basis of a summative grade using the suggested scale below.

Complete presentation submitted 25%

Code with abstractions submitted 25%

Rubric Points (5) 5 * number of rubric points up to 50%

Activity:

Project Development

Wrap up:

As a group, evaluate your progress during this session and identify what still needs to be done.

Session 10. Project Completion

Say: When we do the Create Task you will have 12 hours of class time to design, develop and write a written report about a computer programming topic of you choice. I will not be giving further guidance during those 12 hours so its important to identify any questions that still need to be answered during ti sessions. We have four goals.

- Finish the project
- Finish the written report.
- Submit both project code and written report.
- Identify any remaining questions.

Activity 1 - Finish the project (10 min)

As a group, briefly meet and plan any remaining work needed to complete the prototype.

Complete the development of the prototype.

Activity 2 - Finish the written report (25) min

Evidence of Learning

Formative Assessment

Session 3 ends with students creating their first EarSketch animation. This script should be formatively assessed as all future EArSketch work depends on the ability to use functions like onMeasure() to produce animations.

Session 4 ends with students implementing a change to a data visulization project they built as a class. Tihs project could be the basis of a formative evaluation - perhaps done wile students are developing their designs in the next sessions.

Session 5 ends with students making a plan for a project they will implement in the next session. This plan could be used for fromative assessement.

Summative Assessment

Session 6 Creating your own EarSketch script using input and visualization students implement a project to meet te design submitted yesterday. They also make jounal entries after they assess the process. Both the projet and the journal questions can be used for summative evaluation.

(http://www.umbc.edu/)


(http://www.umd.edu/)


(http://www.nsf.gov/)

Authored by: CS Matters in Maryland
Website: csmatters.org (http://csmatters.org)
Email: csmattersinmaryland@gmail.com (mailto:csmattersinmaryland@gmail.com)